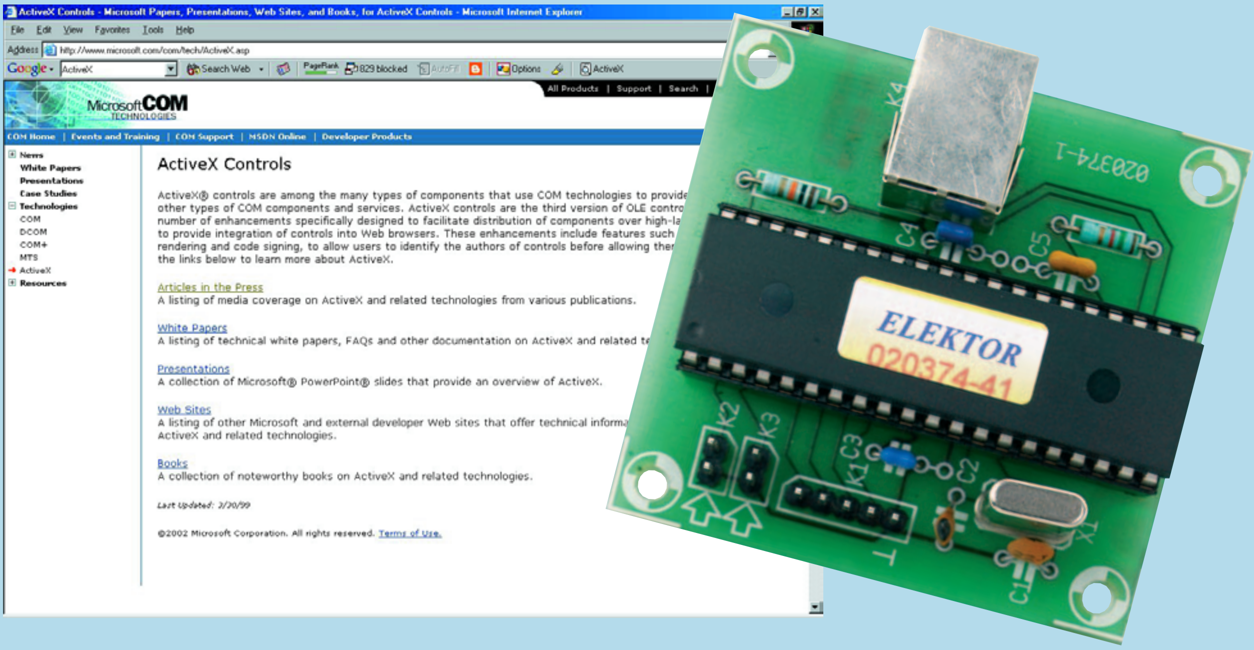
Eddie Brador

Working with ActiveX

ActiveX component for the USB analogue converter


ActiveX Controls - Microsoft Papers, Presentations, Web Sites, and Books, for ActiveX Controls - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://www.microsoft.com/com/tech/ActiveX.asp
Google ActiveX Search Web PageRank 829 blocked AutoFill Options ActiveX
All Products | Support | Search |
Microsoft COM TECHNOLOGIES
COM Home | Events and Training | COM Support | MSDN Online | Developer Products
News
White Papers
Presentations
Case Studies
Technologies
COM
DCOM
COM+
MTS
ActiveX
Resources
K4
020374-1
C4
C5
ELEKTOR
020374-41
K2
K3
K1
C3
C2
T1
X1
C1

ActiveX Controls

ActiveX® controls are among the many types of components that use COM technologies to provide
other types of COM components and services. ActiveX controls are the third version of OLE contro
number of enhancements specifically designed to facilitate distribution of components over high-la
to provide integration of controls into Web browsers. These enhancements include features such
rendering and code signing, to allow users to identify the authors of controls before allowing ther
the links below to learn more about ActiveX.

Articles in the Press
A listing of media coverage on ActiveX and related technologies from various publications.

White Papers
A listing of technical white papers, FAQs and other documentation on ActiveX and related te

Presentations
A collection of Microsoft® PowerPoint® slides that provide an overview of ActiveX.

Web Sites
A listing of other Microsoft and external developer Web sites that offer technical informa
ActiveX and related technologies.

Books
A collection of noteworthy books on ActiveX and related technologies.

Last Updated: 3/20/99

©2002 Microsoft Corporation. All rights reserved. Terms of Use.

The USB analogue converter published in the November 2003 issue is a neat circuit for simple measurement and test purposes. The ActiveX component described here can be used to control this circuit from a high level language.

oMany Elektor Electronics readers constructed this simple circuit, which makes it possible to easily measure analogue voltages and switch digital outputs via USB. Unfortunately, the associated Windows program had a limited functionality and it wasn't always stable. The author was inundated with questions about the program, which led to the development of a utility that allows a user of the USB analogue converter to write a control program for driving any peripheral.

The ActiveX component described here can read in analogue values and drive the digital outputs. This type of control makes it possible to program the circuit from within any high level language, such as Delphi, Visual Basic or C++ by Borland.

Since Delphi is one of the most commonly used programming languages, we'll take a look how we can use this language with the ActiveX component.

What is ActiveX?

An ActiveX component is a piece of software that can be used by a host application that incorporates ActiveX control (such as C++Builder, Delphi, Visual dBASE, Visual Basic, Internet

Explorer or Netscape Navigator), and thereby enhances its functionality.

Delphi comes with several ActiveX components for creating graphs, spreadsheets and pictures. You can add these components to the IDE (Integrated Development Environment) and use them just like any other standard VCL element (*Visual Component Library*) by including them in the program and defining their properties with the help of the *object inspector*.

You can also use an ActiveX component on a web page by including a link to it in an HTML document and displaying it in a browser that supports ActiveX.

The following is a general method for using an ActiveX component in the Delphi programming environment. We won't include all details for writing ActiveX components. For this we refer you to the *Microsoft Developer's Network* (MSDN). More information on ActiveX can also be found on Microsoft's website.

The ActiveX component picus_brxProj1.ocx

The author used the original Basic program as the starting point and continued development in Delphi4. It should be clear that the scope of this article doesn't allow for the inclusion of a complete course on the development of ActiveX components. The Help function in Delphi will provide you with the required information.

The complete installation of the ActiveX component requires another seven files. All these files can be found in the folder **fich_ocx** (shown in the screendump in **Figure 1**).

Installation

We'll now go through the installation steps manually, as this makes the process clearer than an automatic installation.

1. The file picus_brxProj1.ocx should be copied to the folder **c:\windows\system** for Windows 98/ME, to the folder **c:\windows\system32** for Windows 2000 and XP, or to **c:\WINNT\system32** for Windows 2000 Pro.
2. Copy all other files from folder **fich_ocx** to folder **c:\Program Files\Borland\Delphi\Imports** (the exact folder name can vary slightly, depending on the version of Delphi).
3. Click on Start, Run and type in the command regsvr32 picus_brxProj1.ocx (**Figure 2**).
   When this has finished, you should get the message shown in Figure 3.
4. From the *Component* menu in Delphi choose the *Import ActiveX Control* option, then click on *picus_brxProj1*, followed by Install and Save (**Figure 4**).

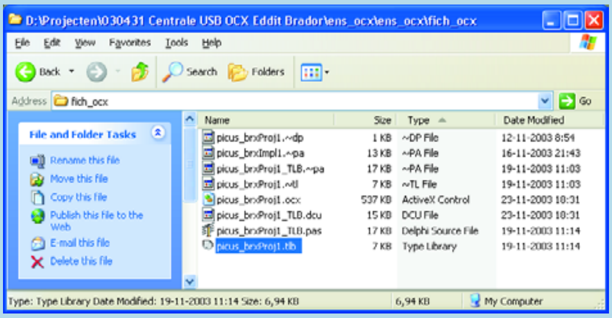
When this has completed satisfactorily, the ActiveX component should become visible.

We can now start writing our own driver program.

An example with the ActiveX component

There is nothing clearer than an example to help understand how things work. We'll start with a very simple one: the temperature measurement using the LM335, a well-known temperature sensor made by National Semiconductor.

Its output voltage is proportional to the temperature. This voltage is amplified by a factor of about six by a TL271. The circuit diagram is shown in **Figure 5**. The supply voltage is derived from one of the digital outputs of the USB analogue converter. Output K2 is connected to the first analogue input on the PCB (also K2).



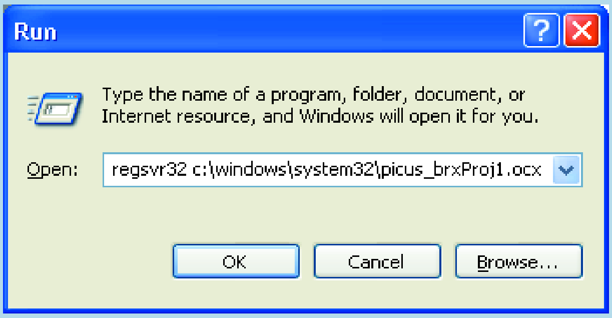
D:\Projecten\030431 Centrale USB OCX Eddit Brador\ens_ocx\ens_ocx\fich_ocx

Address: fich_ocx

Name	Size	Type	Date Modified
picus_brxProj1.~dp	1 KB	~DP File	12-11-2003 8:54
picus_brxImpl1.~pa	13 KB	~PA File	16-11-2003 21:43
picus_brxProj1_TLB.~pa	17 KB	~PA File	19-11-2003 11:03
picus_brxProj1.~tl	7 KB	~TL File	19-11-2003 11:03
picus_brxProj1.ocx	537 KB	ActiveX Control	23-11-2003 10:31
picus_brxProj1_TLB.dcu	15 KB	DCU File	23-11-2003 10:31
picus_brxProj1_TLB.pas	17 KB	Delphi Source File	19-11-2003 11:14
picus_brxProj1.tlb	7 KB	Type Library	19-11-2003 11:14

Type: Type Library Date Modified: 19-11-2003 11:14 Size: 6,94 KB 6,94 KB

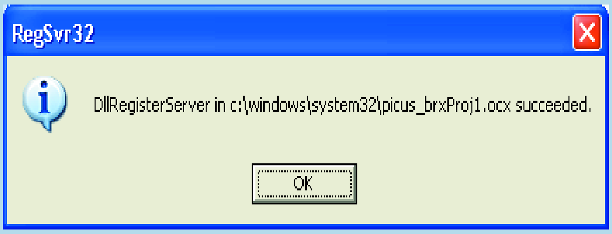
Figure 1. All files for the project are in the folder fich_ocx.



Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: regsvr32 c:\windows\system32\picus_brxProj1.ocx

Figure 2. The Run dialogue box, which everybody must surely have used at some stage.



DllRegisterServer in c:\windows\system32\picus_brxProj1.ocx succeeded.

Figure 3. When this message appears the ActiveX component has been correctly registered.

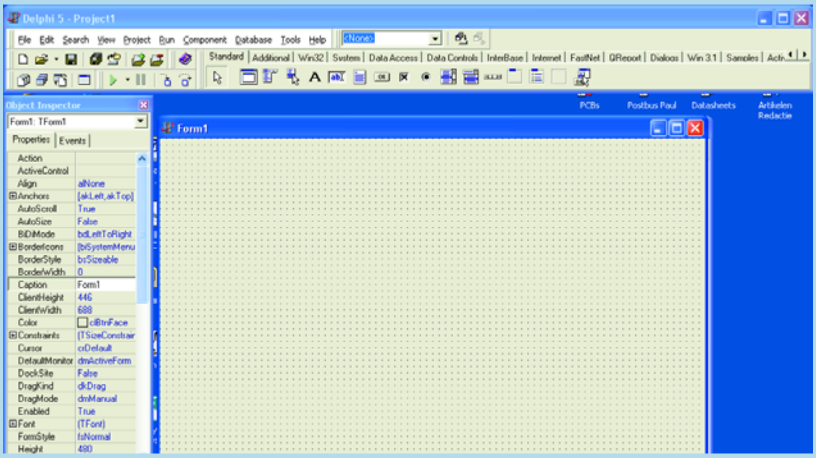
The program

1. Start Delphi, choose File/New Application.
   Click on the ActiveX tab, then click on the component **picus_brx** and put it onto Form1 (**Figure 6**).

2. Put a button (found under the standard elements tab of Delphi) onto the form.
   Then put and edit box (also found under the standard elements tab) onto the form.
   And finally put a timer (found under the system elements tab) onto the form, as shown in **Figure 7**.

3. Double-click on Button1 and enter the following code:

```
procedure TForm1.Button1Click(Sender: TObject);
var Val_T:real;
```


Delphi 5 - Project1
File Edit Search View Project Run Component Database Tools Help
<None>
Standard Additional Win32 System Data Access Data Controls InterBase Internet FastNet QReport Dialogs Win 3.1 Samples ActiveX
Object Inspector
PCBs
Postbus Paul
Datasheets
Artikelen Redactie
Form1
Properties
Events

Form1: TForm1

Action	
ActiveControl	
Align	alNone
⊞Anchors	[akLeft,akTop]
AutoScroll	True
AutoSize	False
BiDiMode	bdLeftToRight
⊞BorderIcons	[biSystemMenu
BorderStyle	bsSizeable
BorderWidth	0
Caption	Form1
ClientHeight	446
ClientWidth	688
Color	☐ clBtnFace
⊞Constraints	(TSizeConstrair
Cursor	crDefault
DefaultMonitor	dmActiveForm
DockSite	False
DragKind	dkDrag
DragMode	dmManual
Enabled	True
⊞Font	(TFont)
FormStyle	fsNormal
Height	480

Figure 4. We can now create our own application.

```
begin
Val_T:=(picus_brx1.AN0*5/255); {U/T conversion}
Edit1.text:=floattostr(val_T);
end;
```

Double-click on the Timer:

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
Button1.Click;
end;
```

When you run the program, you should see a temperature-dependent voltage appear on the screen.

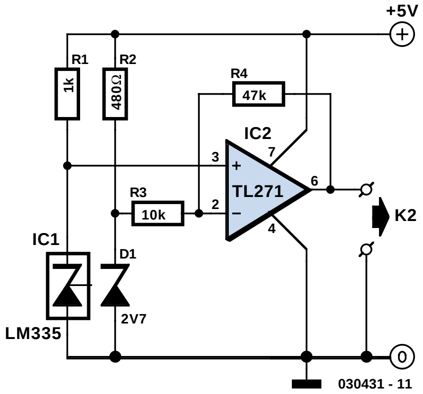


Figure 5. Circuit diagram of a simple thermometer using the well-known LM335.

Using the outputs

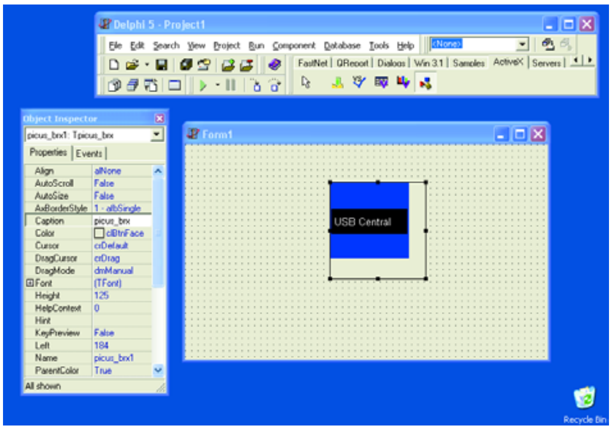
There is a function in the OCX that allows us to control the outputs of the USB analogue converter. We'll use this in the following example.

Open a new form in Delphi and add the following elements:

- 4 Checkbox elements from the Standard library.
- 1 Timer from the System library.
- 1 Button from the Standard library and of course our *picus_brx1* ocx object from the ActiveX library (**Figure 8**).

Double-click on the Form and type in the following code:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
picus_brx1.active;
picus_brx1.visible:=false;
end;
```



picus_brx1: Tpicus_brx

Align	alNone
AutoScroll	False
AutoSize	False
AxBorderStyle	1 - afbSingle
Caption	picus_brx
Color	☐ clBtnFace
Cursor	crDefault
DragCursor	crDrag
DragMode	dmManual
⊞Font	(TFont)
Height	125
HelpContext	0
Hint	
KeyPreview	False
Left	184
Name	picus_brx1
ParentColor	True

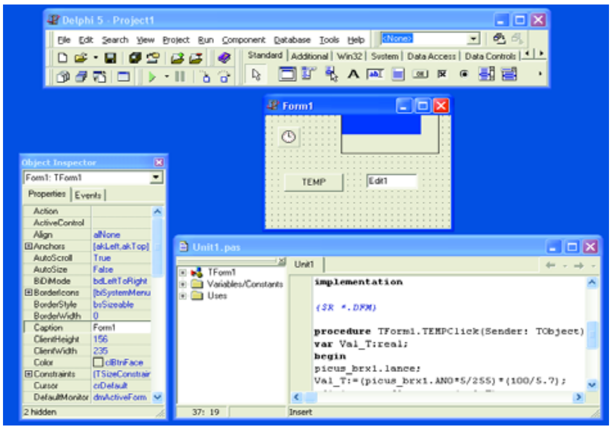
Figure 6. The object 'USB Centrale' has been placed on the form.

Double-click on Button1 and add:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
picus_brx1.J(Ord(Checkbox1.Checked));
picus_brx1.J1(Ord(Checkbox2.Checked));
picus_brx1.J2(Ord(Checkbox3.Checked));
picus_brx1.J3(Ord(Checkbox4.Checked));
picus_brx1.lance;
end;
```

Double-click on the Timer element and add:

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
Button1.Click;
end;
```



Form1: TForm1

Action	
ActiveControl	
Align	alNone
⊞Anchors	[akLeft,akTop]
AutoScroll	True
AutoSize	False
BiDiMode	bdLeftToRight
⊞BorderIcons	[biSystemMenu
BorderStyle	bsSizeable
BorderWidth	0
Caption	Form1
ClientHeight	156
ClientWidth	235
Color	☐ clBtnFace
⊞Constraints	(TSizeConstrair
Cursor	crDefault
DefaultMonitor	dmActiveForm

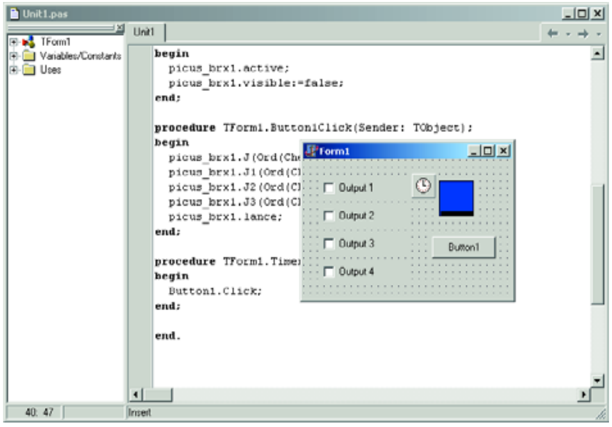
```
implementation

{$R *.DFM}

procedure TForm1.TEMPClick(Sender: TObject)
var Val_T:real;
begin
picus_brx1.lance;
Val_T:=(picus_brx1.AN0*5/255)*(100/5.7);
```

Figure 7. Form1 with Button1, a Timer and an Edit box.

Click on the Checkboxes and see how the module outputs change their state.

These programs only show a few of the many possibilities that are offered by the USB analogue converter in conjunction with the ActiveX control. We'll leave it up to your imagination to come up with some other uses.

(030431-1)



```
begin
  picus_brx1.active;
  picus_brx1.visible:=false;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  picus_brx1.J(Ord(Ch
  picus_brx1.J1(Ord(Cl
  picus_brx1.J2(Ord(Cl
  picus_brx1.J3(Ord(Cl
  picus_brx1.lance;
end;

procedure TForm1.Time
begin
  Button1.Click;
end;

end.
```

Figure 8. An OCX function can be used to test the operation of the outputs.

Web pointers:

ActiveX controls:
www.microsoft.com/com/tech/ActiveX.asp
JEDI Visual Component Library:
http://homepages.borland.com/jedi/jvcl/
VCLComponents.com:
www.vclcomponents.com/
Author's website (under construction):
http://dorelec.web.oo.fr